



STUDY OF QUALIFICATION CRITERIA FOR SOFTWARE VERIFICATION TOOLS

A report presented at the
2005 National Software & Complex Electronic
Hardware Standardization Conference

27-July-2005

Vdot Santhanam
PRINCIPAL INVESTIGATOR
BOEING

TOPICS

- Overview of Project
- Phase 1 Summary
- Phase 2 Summary
- Phase 3 Activities
 - Test Suite Framework
 - Model Test Suite
 - Results
 - Summary
- Recommendations for Future Work

Overview of Project

- Objective of Study
 - Investigate qualification criteria for software structural coverage verification tools
 - Determine whether the regulatory guidance provides sufficient basis for determining whether an automated verification tool enforces the DO-178B coverage criteria accurately
 - Recommend means for improving the objectivity and uniformity of tool qualification process
- Study organized into three phases
 - Phase 1 – Research the issues
 - Phase 2 – Study and recommend means to address the issues
 - Phase 3 – Demonstrate the efficacy of recommendations

Phase 1 Findings

- Phase 1 of the project found that
 - The current regulatory guidance to be the source of many ambiguities
 - Ambiguities allowed tool vendors and regulatory authorities to interpret coverage criteria in varied ways
 - The study surveyed twenty-one tools from nineteen vendors
 - Most offered coverage analysis per DO-178B levels A, B, and C
 - The basis for an objective set of qualification criteria should begin by clarifying the DO-178B intent
 - The feasibility of developing a test suite to improve objectivity should be investigated

Phase 2 Findings

- Phase 1 identified issues dealing with the interpretation of DO-178B structural coverage criteria
 - Statement Coverage
 - Decision Coverage
 - MC/DC
- Phase 2 studied ways to resolve these issues and made recommendations
- Phase 2 also found that a test suite to bring about uniform interpretation is feasible

Statement Coverage Issues

- Statement Coverage Issues
 - Should implicit statements be subject to coverage?
 - Recommendation: No.
 - Should declarative statements be subject to coverage?
 - Recommendation: Yes, if the declaration generates executable object code.

Decision Coverage Issues (1 of 2)

- Decision Coverage Issues
 - What is a *Decision*?
 - Recommendation: Binary valued expressions that are: (a) declared as Boolean, or (b) interpreted as Boolean in one or more contexts, or (c) derive their values from other such expressions
 - How are Boolean constants to be treated?
 - Recommendation: Not subject to coverage
 - How are exception handlers to be treated with respect to entry and exit coverage?
 - Recommendation: Each handler should be subject entry/exit coverage and statements in it subject to decision coverage

Decision Coverage Issues (2 of 2)

- Decision Coverage Issues
 - What are the contexts in which Boolean expressions should be subject to decision coverage?
 - Recommendations:
 - Should only apply to those that appear in branching constructs
 - Should be renamed as *Branch Coverage*
 - The definition of a decision as any Boolean expression should be retained for MC/DC purposes
 - This is in contrast to recommendations of CAST-10 Position Paper
 - Presented the new recommendations and rationale at the CAST meeting, Seattle, July 2004.

MC/DC Issues (1 of 3)

- Modified Condition/Decision Coverage Issues
 - How should decisions containing short-circuit operators be treated
 - Recommendation: Treat each short-circuited term as an independent, top-level decision, in harmony with the flow graph model suggested by DO-248B
 - The study also noted that this does not mean that branch coverage for Boolean expressions containing arbitrary set of short-circuit operators is equivalent to MC/DC

MC/DC Issues (2 of 3)

- What are *conditions*, *decisions*, *Boolean operators*?

- Recommended definitions:

Condition: A lowest-level Boolean expression that is:

- (a) A Boolean variable (including array element and record component), OR
- (b) a Boolean function call, OR
- (c) an expression consisting of non-Boolean terms and predefined operators, delivering a Boolean result

Boolean operator: An operator that operates on one or more Boolean operands and delivers a Boolean result

Decision: (a) A condition, OR (b) the result of a Boolean operator

MC/DC Issues (3 of 3)

- How is the apparent contradiction in MC/DC as it applies to decisions containing coupled (replicated) conditions to be resolved?
 - DO-178B states that each *occurrence* of a condition must be treated as a *separate* condition
 - However, to show independence, each condition must be toggled while holding all other conditions fixed
 - Tackling this issue was the single greatest challenge for the study

Resolving MC/DC Contradiction

- Phase 2 investigated eight variants of MC/DC
 - Five that are different interpretations of the DO-178B definition of MC/DC
 - These variants are referred to here as “flavors”
 - Three that could be considered alternate forms of Boolean expression structural coverage
 - These variants are referred to as “alternates”

Resolving MC/DC Contradiction (contd.)

- The study concluded
 - Among the five flavors of DO-178B definition
 - [UCM] Unique-Cause MC/DC is the simplest, but not applicable to decisions containing coupled conditions
 - [MSM] Masking MC/DC is the most widely applicable and the most complex
 - [CCM] Coupled-Cause MC/DC is as widely applicable as MSM, but is weaker than MSM

Resolving MC/DC Contradiction (contd.)

- The study also concluded
 - Among the three alternate forms of Boolean expression coverage
 - [OCC] Operator Coverage Criterion most closely matches the intent of DO-178B and is significantly simpler to describe and implement
 - OCC is weaker than MSM, but does that matter? More research is needed..

MC/DC Issues (3 of 3, contd.)

- The study recommended
 - For the near-term, accept MSM or CCM as meeting the DO-178B requirement
 - For DO-178C, study alternate forms of structural coverage to replace MC/DC
 - Recommend OCC as a starting point

Phase 3 Activities

- Phase 3 activities consisted of
 - Formulating a test suite
 - Defining test objectives
 - Constructing a prototype test suite
 - Running the tests against selected tools
 - Making recommendations on the development of a full-scale test suite

Test Suite Objectives

- Primary Objectives for a Tool Qualification Test Suite
 - Should be applicable to tools for popular languages, with ability to exercise language-specific constructs
 - Should be applicable to tools used to verify application software at level A, B, or C
 - Should be tailorable to multiple tools, and multiple compilers (if tool is compiler-independent)
 - Should minimize manual activity required to run the test suite against a given tool

Test Suite Formulation

- Boeing has constructed a framework for a test suite, called CATS-178B, that meets the primary objectives
 - Addresses coverage at all three levels of criticality
 - Is largely language independent, with ability to include language-specific tests
 - Is largely tool independent, with tailorable scripts to invoke tool and compiler

Test Suite Formulation (contd.)

- CATS is organized by criticality level of the software to be verified by the tool
 - Level C: Statement Coverage
 - Level B: Decision or Branch Coverage
 - Level A: (Flavors of) MC/DC
- A tool needing qualification at a given level must pass all tests applicable to the lower level(s)

Test Suite Formulation (contd.)

- At each level CATS includes affirmative and negative tests
 - Affirmative tests
 - Confirm the tool correctly reports coverage as attained from a given set of test cases
 - Negative tests
 - Confirm the tool correctly reports coverage as deficient from a given set of test cases
 - Discriminating tests
 - Determine the specific interpretation used by the tool from among acceptable alternate interpretations
 - Failure of a tool on a negative test is more serious than a failure on an affirmative test

Test Suite Formulation (contd.)

- Tests in CATS are based on test objectives that vary by
 - the level of criticality for which the tool is to be qualified
 - the degree of sophistication of the tests
 - Basic tests: expect all tools to pass without difficulty or variation in results
 - Advanced tests: expect a tool to pass unless the tool's limitations are clearly documented and the application will not violate them

Test Suite Prototype

- CATS/p
 - Uses a Test Description Language specifically designed to allow generic test descriptions
 - Includes a cross-section of tests from all levels
 - 27 Level C, 23 Level B, 115 Level A
 - Includes Affirmative, Negative or Discriminator tests
 - Includes an Indeterminate class for tests whose validity will depend on of resolution of coverage issues identified in phase 2
 - Includes tests to exercise Ada-unique constructs
 - Includes customizable scripts to generate drivers and invoke the tool on them

Phase 3 Results

- Feasibility of a Test Suite that could be used to improve objectivity and uniformity in tool qualification was demonstrated
 - A test suite framework was formulated to meet all major objectives
 - A model test suite was written based on that framework to demonstrate the feasibility
 - The model was validated against three tools
 - Referred to as Tools A, B, and C for anonymity

Tool A Results

Test Level	Pass	False Negative	False Positive	FN-Tool Limita-tion	FP-Tool Limita-tion	No Data	Others	Total
C	22		4		1			27
B	14	1	5		3			23
A	84	10				7	14	115

27-Jul-2005

Tool B Results

Test Level	Pass	False Negative	False Positive	FN-Tool Limita-tion	FP-Tool Limita-tion	No Data	Others	Total
C	22	2	3					27
B	3	14	5		1			23
A	50	10	6			19	30	115

27-Jul-2005

Tool C Results

Test Level	Pass	False Negative	False Positive	FN-Tool Limita-tion	FP-Tool Limita-tion	No Data	Others	Total
C	22						5	27
B	18		4				1	23
A	82	3	19		1	9	1	115

27-Jul-2005

CATS/p Preliminary Results

- Summary of preliminary results
 - The test suite was effective in finding significant deficiencies in all three tools
 - Tests were found to be portable across tools
 - Output reports had to be manually reviewed due to lack of uniform presentation
 - We believe that vendors will address the deficiencies if validation were required, leading to more uniform interpretation

Recommendations for Future Work

- The development of a full-scale test suite should be launched
 - The use of a standard test suite to validate coverage analysis will bring objectivity and uniformity to the tool qualification process
 - The test suite could serve as a “final authority” in resolving ambiguities in any natural language statement of the requirements